DYNAMIC FRAME-SKIPPING IN VIDEO TRANSCODING

Jenq-Neng Hwang and Tzong-Der Wu
Information Processing Lab.
Department of Electrical Engineering
Box 352500, University of Washington
Seattle, WA 98195
Email: hwang@ee.washington.edu

Chia-Wen Lin
Customer Premises Equipment &
Access Technology Department
Computer & Communication Lab.
Industrial Technology Research Lab.
195-11 Sec. 4, Chung Hsing Rd.,
Chutung, Hsinchu, Taiwan 310, R.O.C.
Email: ljw@n100.ccl.itri.org.tw

Abstract - This paper investigates the dynamic frame skipping strategy in video transcoding. To speed up the operation, a video transcoder usually reuses the decoded motion vectors to reencode the video sequences at a lower bit-rate. When frame skipping is allowed in a transcoder, those motion vectors can not be reused because the motion vectors of the current frame is no longer estimated from the immediate past frame. To reduce the computational complexity of motion vectors reestimation, a bilinear interpolation approach is developed to overcome this problem. Based on these interpolated motion vectors, the search range can be much reduced. Furthermore, we propose a frame rate control scheme which can dynamically adjust the number of skipped frames according to the accumulated magnitude of motion vectors. As a result, the decoded sequence can present much smoother motion.

1. INTRODUCTION

For the lower bit-rate video compression standards, such as H.261 [1] and H.263 [2], the frames could be skipped to keep the generated bit-rate from exceeding the allocated channel bandwidth. The video transcoder is a device which converts a compressed format into another compressed format, specifically the bit-rate reduction problem discussed in this paper. In the proceeding of transcoding, if the bandwidth of the outgoing channel is not enough to allocate bits with requantization, the frame skipping is a good strategy to control the bit-rate and at the same time to keep the image quality within an acceptable level. Usually in a transcoder, the motion vectors decoded from the bit streams are reused to speed up the reencoding process [3], where the frames can not be skipped because motion vectors of each frame are estimated from its immediate past frame. If frame skipping is necessary, the frame must be decompressed completely and the motion estimation must be performed again to search the motion vectors, which can create undesirable problems in the real-time transcoding.

In this paper, a bilinear interpolation method is developed to estimate the motion vectors from the current frame to the previous non-skipped frame given the motion vector between every adjacent frame are known. The newly located position based on this interpolated motion vector will serve as the new search center, thus the search range can be much reduced. In addition, for the frame rate control problem, randomly skipping frames based on consumed bits of the last encoded frame can cause abrupt motion. Taking advantage of known motion vectors, the proposed dynamic frame skipping scheme dynamically estimate the motion activities of each frame so that the length of skipped frames can be determined by an optimal way.

2. INTERPOLATION OF MOTION VECTORS

In video transcoding, usually the motion vectors decoded from the bitstream are reused to speed up the reencoding process. Because only the motion between adjacent frames is known, the motion estimation for the non-skipped frames requires the path of motion from the current frame to its previously non-skipped frame, which could be many frames away.

If the motion vectors between every adjacent frames are known, the problem of tracing the motion from frame 4 to frame 1 in Figure 1 could be partly solved by using the repeated applications of bilinear interpolation [4].

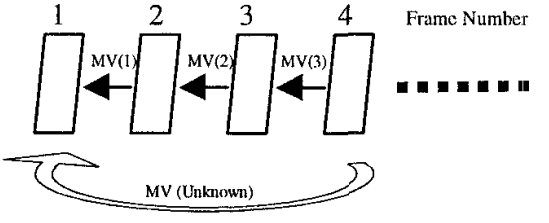


Figure 1. A Motion Tracing Example.

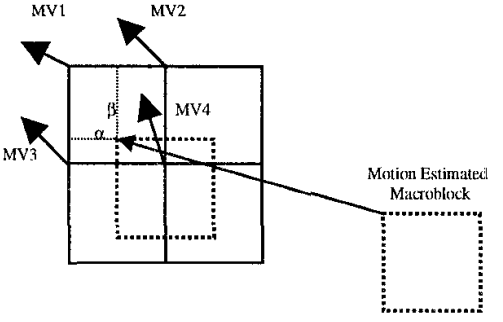


Figure 2. Interpolation of Motion Vectors.

Referring to Figure 2, a shifted macroblock is located in the middle of four neighbor macroblocks. The bilinear interpolation is defined as:

$$MV_{\text{int}} = (1-\alpha)(1-\beta)MV_1 + (\alpha)(1-\beta)MV_2 + (1-\alpha)(\beta)MV_3 + (\alpha)(\beta)MV_4, \quad (1)$$

where $MV_1 \sim MV_4$ are the motion vectors of the four neighbor macroblocks. α and β are determined by the pixel distance to MV_1. The weighting constant of each neighbor block is inversely proportional to the pixel distance. Repeatedly tracing the motion in this manner will create an extended motion vector for each macroblock in the current frame to its previously non-skipped frame.

3. DYNAMIC SEARCH RANGE ADJUDGEMENT

The bilinear interpolation method can partly solve the motion vectors reusing problem, however, further adjustment of the reestimated motion vectors has to be performed by using a smaller search range. For each macroblock, we use the new position located by the interpolated and composited motion vectors to be the search center for the final motion reestimation stage. We use the notation, MV_{max}, to define the maximum search distance described in most video compression standards. If the horizontal and vertical directions of the composited motion vector for macroblock i is MV_X_i and MV_Y_i, then the search distance D_i to the new search center for each macroblock i can be described by the following pseudo codes.

For i = 1 to Maximum Number of Macroblocks
$D_i = MV_{max} - MAX(|MV\_X_i|, |MV\_Y_i|)$,

The search range can be further reduced by the following approach. The correctness of the composited motion vectors depends on the number of skipped frames and the accumulated magnitudes of the motion vectors. Based on these two factors, we can globally adjust the search range for each coded frame. We also locally adjust the search range for each macroblock by referring to the magnitude of the interpolated motion vectors. Assuming in the encoder buffer, the maximum number of skipped frames is N_{max} and the maximum accumulated magnitude of motion vectors is M_{max}. Further assume the number of skipped frames is N_{skp} and the accumulated magnitude of motion vectors between two unskipped frames is M_{acc}, then the new search distance is

$$D\_NEW_i = (w1 \times ADJ_{(frame)} + w2 \times ADJ_{i(mb)})D_i, \quad (w1 + w2 = 1),$$
$$ADJ_{(frame)} = wg1\frac{N_{skp}}{N_{\max}} + wg2\frac{M_{acc}}{M_{\max}}, \quad (wg1 + wg2 = 1),$$
$$ADJ_{i(mb)} = \frac{MV_i}{MV_{\max}}, \qquad (2)$$

where $ADJ_{(frame)}$ is the adjustment for each encoded frame and $ADJ_{i(mb)}$ is the adjustment for each macroblock. wg_1 and wg_2 are the weighting constants for the frame level adjustment as well as w_1 and w_2 are the weighting constants for the overall adjustment. Using this method, the search range can be further reduced and the searching time can thus be cut down.

4. DYNAMIC FRAME SKIPPING FOR FRAME RATE CONTROL

After solving the motion reestimaiton problem, we develop a strategy for determining the length of the skipped frames. The goal is to make the motion of the decoded sequence smoother. Since the known motion vectors serve as a good indication for frame skipping, we can preset a threshold and if the accumulated magnitude of motion vectors after an unskipped frame exceeds this threshold, this frame will be encoded. The threshold is determined by using the number of frames to divide the accumulated magnitude of motion vectors in a buffer. The

threshold is recursively reset after transcoding each frame because the number of encoded frames should be dynamically adjusted according to the variation of the generated bits when the last unskipped frame is transcoded. When moving to the next run of buffer, the remaining bits should be compensated to the current buffer. The whole procedure can be described by the following pseudo codes.

```
While (next buffer) {
   bits = target bits per buffer + compensated bits
   do {
        no. of frames = bits/bits per frame
        threshold = max. accu. mv / no. of frames
        if(accu. mv > threshold) {
           frame is coded
           bits = bits – generated bits}
     }while (bits < bits per frame)
},
```

5. SIMULATION RESULTS

Table 1 shows the percentage of the transcoding time as well as the generated bits of using our new reduced search motion reestimation approach v.s. the standard full search motion reestimation approach. Because the search range is dynamically reduced, the average reencoding time is half of the original coding time, while maintaining the same compression ratio. Table 2 shows the PSNR by using the original and new approaches described above. As indicated by these data, the interpolated motion vectors fine-tuned by using small search range produce negligible image quality degradation.

Sequences (400 Frames)	Transcoding Time (% of Original Algorithm)	Generated Bits (% of Difference)
Foreman	58.97%	4.52%
Grandma	40.42%	4.09%
Mthr_dotr	46.71%	3.05%
Salesman	41.25%	1.23%

Table 1. Transcoding Time and Generated Bits

Sequences (400 Frames)	Average PSNR (OriginalAlgorithm)	Average PSNR (New Algorithm)	Difference
Foreman	33.75	33.73	-0.02
Grandma	33.31	33.27	-0.04
Mthr_dotr	33.52	33.49	-0.03
Salesman	31.58	31.60	+0.02

Table 2. PSNR Differences

As shown in Figure 4, two video sequences (400 frames of Foreman and Grandma) are transcoded by using two approaches. The solid lines are the results of using the standard full search motion reestimation algorithm and the '+' lines are the results of using new approach. We can see these two lines are almost overlapped, so from the picture quality point of view, these two approaches generate the equal quality of images, but the coding time is much reduced. Both simulations used the frame skipping scheme in TMN5 [5], where the frame skipping is determined by how much the consumed bits of the last frame exceed the average target bits per frame.

The comparative results of using TMN5 frame skipping and our dynamic frame skipping scheme are shown in Figure 5, solid-lines present the motion activities of foreman sequence from frame 200 to frame 300 and the dot-lines present the coded frame number when sequence is transcoded from 128 kb/s to 64kb/s. The results shown in the upper plot indicate that the frames are skipped in a random way when using TMN5 skipping scheme, however, using our dynamic skipping scheme as shown in the lower plot, the frames are skipped following the motion activities of each frame, therefore, the video displayed at the receiver site are smoother. This results can also be proved in Figure 6 and Figure 7.

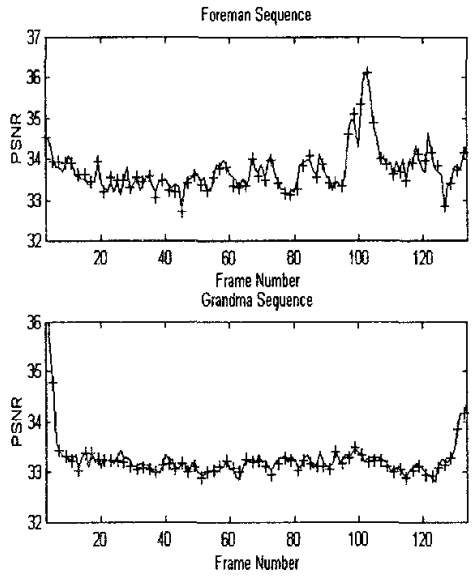


Figure 4. PSNR Plots of Two Sequences by Using Two Search Schemes
Solid line: Using Full Search Scheme
'+' line: Using New Scheme

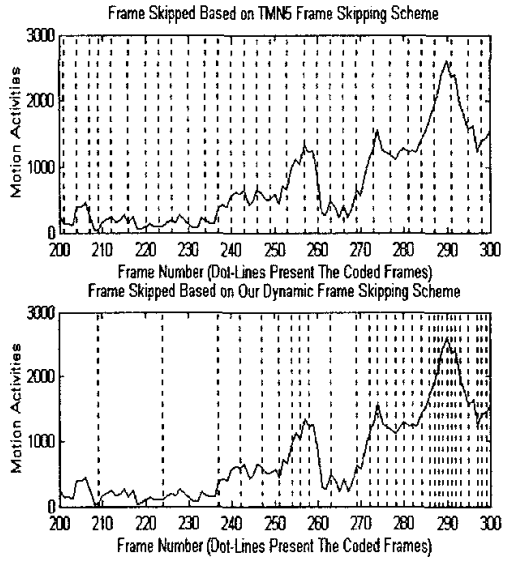


Figure 5 Coded Frames Number After Using TMN5 Frame Skipping Scheme and Dynamic Frame Skipping Scheme


Fm 287 Fm 291
(Two Frames Are Coded: Using Frame Skipping Scheme in TMN5)

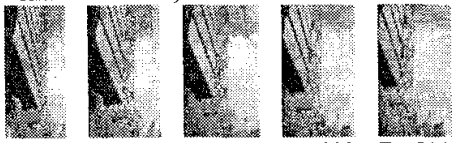
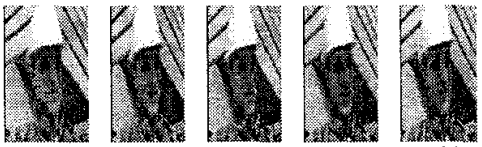

Fm 287 Fm 288 Fm 289 Fm 290 Fm 291
(Five Frames Are Coded: Using Our Dynamic Frame Skipping Scheme)

Figure 6 Using Two Frame Skipping Scheme in Active Frames


Fm 201 Fm 204 Fm 207 Fm 209 Fm 212
(Five Frames Are Coded: Using Frame Skipping Scheme in TMN5)


Fm 209
(One Frame Is Coded: Using Our Dynamic Frame Skipping Scheme)

Figure 7 Using Two Frame Skipping Schemes in Motionless Frames

As shown in Figure 6, objects from frame 287 to frame 291 are active: the man moves out from the frame and the background shifts to the left side. When using the TMN5 skipping scheme as shown in upper line, only two frames are coded, therefore, the man disappears abruptly. However, if using the dynamic skipping scheme as shown in lower line, five frames are coded, therefore, the man moves out smoothly and the background shifts to the left side gradually. Figure 7 shows the encoded frames from frame 200 to frame 215 by using two frame skipping approaches. The objects in these frames are motionless. The TMN5 skipping scheme codes all these five similar frames but the dynamic scheme just codes one frame to represent these five frames, thus the consumed bits are much reduced. Because the saved bits from motionless frames can be used in active frames, the picture quality of the active frames is thus much improved.

6. DISCUSSION AND CONCLUSION

This paper investigates the possibility of interpolating the motion vectors when frames are skipped. Simulation results show that the average coding time is half of the original coding time, but the image qualities and bit rates remain intact. This new approach provides a fast method to implement the transcoder when the frames need to be skipped. The frame rate control problem is also discussed in this paper. The simulation results show that the dynamic frame skipping scheme developed in this paper not only smooth the motion in the decoded sequence, but also dynamically adjust the picture qualities according to the motion activities of each frame.

References

[1] ITU-T Draft Recommendation H.261, "Video codecs for audiovisual services at p x 64 kb/s," May 1992.

[2] ITU-T Draft Recommendation H.263, "Video coding for narrow telecommunication channels," June 1995.

[3] G. Keeman, R. Hellinghuizen, F. Hoeksema and G. Heideman, "Transcoding of MPEG-2 bitstreams," *Signal Processing: Image Communication*, Vol. 8, pp. 481-500, September 1996.

[4] Austin Lan and J.N.Hwang, "Context Dependent Reference Frame Placement for MPEG Video Coding," *IEEE ICASSP'97*, Vol. 4, pp. 2997-3000, April, 1997.

[5] ITU Study Group 15, Working Party 15/1, "Video Codec Test Model Model (TMN5)," Jan. 1995

[6] M.T.Sun, T.D.Wu and J.N.Hwang, "Dynamic Bit Allocation in Video Combining for Multipoint Conferencing," *IEEE Trans. Circuits and Systems*, Vol. 45, no. 5, pp. 644-648, May 1998.